

# Cloudera

## Exam CCA-500

### Cloudera Certified Administrator for Apache Hadoop (CCAHA)

Version: Demo

[ Total Questions: 10 ]

**Question No : 1**

Given:

```
[user1@host1 ~] yarn application -list
Total Applications: 3
Application ID      Application-Name  Application-Type  User  Queue  State  Final-State  Progress  Tracking
Application_1374638600275_0109  Sleep Job      MAPREDUCE        user1  KILLED  KILLED  KILLED       100%     host1:54059
Application_1374638600275_0121  Sleep Job      MAPREDUCE        user1  FINISHED  SUCCEEDED  SUCCEEDED    100%     host1:19888/Jobhistory/Job_1374638600275_0121
Application_1374638600275_0020  Sleep Job      MAPREDUCE        user1  FINISHED  SUCCEEDED  SUCCEEDED    100%     host1:19888/Jobhistory/Job_1374638600275_0020
```

You want to clean up this list by removing jobs where the State is KILLED. What command you enter?

- A. Yarn application –refreshJobHistory
- B. Yarn application –kill application\_1374638600275\_0109
- C. Yarn radmin –refreshQueue
- D. Yarn radmin –kill application\_1374638600275\_0109

**Answer: B**

Reference: [http://docs.hortonworks.com/HDPDocuments/HDP2/HDP-2.1-latest/bk\\_using-apache-hadoop/content/common\\_mrv2\\_commands.html](http://docs.hortonworks.com/HDPDocuments/HDP2/HDP-2.1-latest/bk_using-apache-hadoop/content/common_mrv2_commands.html)

**Question No : 2**

You are running a Hadoop cluster with MapReduce version 2 (MRv2) on YARN. You consistently see that MapReduce map tasks on your cluster are running slowly because of excessive garbage collection of JVM, how do you increase JVM heap size property to 3GB to optimize performance?

- A. yarn.application.child.java.opts=-Xsx3072m
- B. yarn.application.child.java.opts=-Xmx3072m
- C. mapreduce.map.java.opts=-Xms3072m
- D. mapreduce.map.java.opts=-Xmx3072m

**Answer: C**

Reference: <http://hortonworks.com/blog/how-to-plan-and-configure-yarn-in-hdp-2-0/>

**Question No : 3**

You have installed a cluster HDFS and MapReduce version 2 (MRv2) on YARN. You have no `dfs.hosts` entry(ies) in your `hdfs-site.xml` configuration file. You configure a new worker node by setting `fs.default.name` in its configuration files to point to the NameNode on your cluster, and you start the `DataNode` daemon on that worker node. What do you have to do on the cluster to allow the worker node to join, and start sorting HDFS blocks?

- A. Without creating a `dfs.hosts` file or making any entries, run the commands `hadoop.dfsadmin-refreshModes` on the NameNode
- B. Restart the NameNode
- C. Creating a `dfs.hosts` file on the NameNode, add the worker Node's name to it, then issue the command `hadoop dfsadmin -refresh Nodes =` on the Namenode
- D. Nothing; the worker node will automatically join the cluster when NameNode daemon is started

**Answer: A**

**Question No : 4**

Your cluster implements HDFS High Availability (HA). Your two NameNodes are named `nn01` and `nn02`. What occurs when you execute the command: `hdfs haadmin -failover nn01 nn02`?

- A. `nn02` is fenced, and `nn01` becomes the active NameNode
- B. `nn01` is fenced, and `nn02` becomes the active NameNode
- C. `nn01` becomes the standby NameNode and `nn02` becomes the active NameNode
- D. `nn02` becomes the standby NameNode and `nn01` becomes the active NameNode

**Answer: B**

**Explanation: Explanation:**

`failover-` initiate a failover between two NameNodes

This subcommand causes a failover from the first provided NameNode to the second. If the first

NameNode is in the Standby state, this command simply transitions the second to the Active state without error. If the first NameNode is in the Active state, an attempt will be made to gracefully transition it to the Standby state. If this fails, the fencing methods (as configured by `dfs.ha.fencing.methods`) will be attempted in order until one of the methods succeeds. Only after this process will the second NameNode be transitioned to the Active state. If no fencing method succeeds, the second NameNode will not be transitioned to the Active state, and an error will be returned.

**Question No : 5**

Assuming a cluster running HDFS, MapReduce version 2 (MRv2) on YARN with all settings at their default, what do you need to do when adding a new slave node to cluster?

- A.** Nothing, other than ensuring that the DNS (or/etc/hosts files on all machines) contains any entry for the new node.
- B.** Restart the NameNode and ResourceManager daemons and resubmit any running jobs.
- C.** Add a new entry to /etc/nodes on the NameNode host.
- D.** Restart the NameNode of dfs.number.of.nodes in hdfs-site.xml

**Answer: A**

**Explanation:**

[http://wiki.apache.org/hadoop/FAQ#I\\_have\\_a\\_new\\_node\\_I\\_want\\_to\\_add\\_to\\_a\\_running\\_Hadoop\\_cluster.3B\\_how\\_do\\_I\\_start\\_services\\_on\\_just\\_one\\_node.3F](http://wiki.apache.org/hadoop/FAQ#I_have_a_new_node_I_want_to_add_to_a_running_Hadoop_cluster.3B_how_do_I_start_services_on_just_one_node.3F)

**Question No : 6**

You have recently converted your Hadoop cluster from a MapReduce 1 (MRv1) architecture to MapReduce 2 (MRv2) on YARN architecture. Your developers are accustomed to specifying map and reduce tasks (resource allocation) tasks when they run jobs: A developer wants to know how specify to reduce tasks when a specific job runs. Which method should you tell that developers to implement?

- A.** MapReduce version 2 (MRv2) on YARN abstracts resource allocation away from the idea of “tasks” into memory and virtual cores, thus eliminating the need for a developer to specify the number of reduce tasks, and indeed preventing the developer from specifying the number of reduce tasks.
- B.** In YARN, resource allocations is a function of megabytes of memory in multiples of 1024mb. Thus, they should specify the amount of memory resource they need by executing `-D mapreduce-reduces.memory-mb-2048`
- C.** In YARN, the ApplicationMaster is responsible for requesting the resource required for a specific launch. Thus, executing `-D yarn.applicationmaster.reduce.tasks=2` will specify that the ApplicationMaster launch two task contains on the worker nodes.

**D.** Developers specify reduce tasks in the exact same way for both MapReduce version 1 (MRv1) and MapReduce version 2 (MRv2) on YARN. Thus, executing `-D mapreduce.job.reduces=2` will specify reduce tasks.

**E.** In YARN, resource allocation is function of virtual cores specified by the ApplicationManager making requests to the NodeManager where a reduce task is handled by a single container (and thus a single virtual core). Thus, the developer needs to specify the number of virtual cores to the NodeManager by executing `-p yarn.nodemanager.cpu-vcores=2`

**Answer: D**

**Question No : 7**

You're upgrading a Hadoop cluster from HDFS and MapReduce version 1 (MRv1) to one running HDFS and MapReduce version 2 (MRv2) on YARN. You want to set and enforce version 1 (MRv1) to one running HDFS and MapReduce version 2 (MRv2) on YARN. You want to set and enforce a block size of 128MB for all new files written to the cluster after upgrade. What should you do?

**A.** You cannot enforce this, since client code can always override this value

**B.** Set `dfs.block.size` to 128M on all the worker nodes, on all client machines, and on the NameNode, and set the parameter to final

**C.** Set `dfs.block.size` to 128 M on all the worker nodes and client machines, and set the parameter to final. You do not need to set this value on the NameNode

**D.** Set `dfs.block.size` to 134217728 on all the worker nodes, on all client machines, and on the NameNode, and set the parameter to final

**E.** Set `dfs.block.size` to 134217728 on all the worker nodes and client machines, and set the parameter to final. You do not need to set this value on the NameNode

**Answer: C**

**Question No : 8**

On a cluster running CDH 5.0 or above, you use the `hadoop fs -put` command to write a 300MB file into a previously empty directory using an HDFS block size of 64 MB. Just after this command has finished writing 200 MB of this file, what would another user see when they look in directory?

**A.** The directory will appear to be empty until the entire file write is completed on the cluster

**B.** They will see the file with a `._COPYING_` extension on its name. If they view the file,

they will see contents of the file up to the last completed block (as each 64MB block is written, that block becomes available)

**C.** They will see the file with a `._COPYING_` extension on its name. If they attempt to view the file, they will get a `ConcurrentFileAccessException` until the entire file write is completed on the cluster

**D.** They will see the file with its original name. If they attempt to view the file, they will get a `ConcurrentFileAccessException` until the entire file write is completed on the cluster

**Answer: B**

**Question No : 9**

A slave node in your cluster has 4 TB hard drives installed (4 x 2TB). The DataNode is configured to store HDFS blocks on all disks. You set the value of the `dfs.datanode.du.reserved` parameter to 100 GB. How does this alter HDFS block storage?

**A.** 25GB on each hard drive may not be used to store HDFS blocks

**B.** 100GB on each hard drive may not be used to store HDFS blocks

**C.** All hard drives may be used to store HDFS blocks as long as at least 100 GB in total is available on the node

**D.** A maximum of 100 GB on each hard drive may be used to store HDFS blocks

**Answer: C**

**Question No : 10**

You are running Hadoop cluster with all monitoring facilities properly configured.

Which scenario will go undetected?

**A.** HDFS is almost full

**B.** The NameNode goes down

**C.** A DataNode is disconnected from the cluster

**D.** Map or reduce tasks that are stuck in an infinite loop

**E.** MapReduce jobs are causing excessive memory swaps

**Answer: B**

**Thank You For Trying CCA-500 PDF Demo**

**To try our CCA-500 Premium Files visit link below:**

**<https://examsland.com/latest-exam-questions/CCA-500/>**

**Start Your CCA-500 Preparation**

**Use Coupon **EL25** for extra 25% discount on the purchase of Practice Test Software.**